DD2445 COMPLEXITY THEORY: LECTURE 14

Last two lectures

Interlude about communication complexity
Extremely useful in many areas of TCS

Before the autumn break

Randomized computation

Probabilistic Turing machine (PTM)
Flips fair coin at each step

BPP Polynomial time; get answer right with probability $\geq 2/3$
(can be boosted to exponentially close to 1)

One-sided error

RP: No false positives — answer $x \in L$ always right

coRP: No false negatives — answer $x \notin L$ always right

"Zero-sided error"

ZPP Expected polynomial time
If terminates with answer, then always correct

$P \subseteq BPP \subseteq EXP$ [Believe $P = BPP$]

$BPP \subseteq P/poly$

boost success probability $\Rightarrow$
"golden" random string that always works $\Rightarrow$
use as advice

$BPP \subseteq \Sigma_2^P \cap \Pi_2^P$ $(\subseteq PH)$ — didn't show

Complete problems ? Hierarchy theorems ?

- Seems hard; BPP semantic class

Randomized reductions

$L \leq_r L'$ if $\exists$ poly-time PTM M s.t.

$\forall x \in \{0,1\}^*$ $Pr[L'(M(x)) = L(x)] \geq 2/3$

If $L' \in BPP$ and $L \leq_r L'$ then $L \in BPP$

But randomized reductions are not transitive

$\boxed{BP \cdot NP} = \{ L \mid L \leq_r 3\text{-SAT} \}$

What is a PROOF?

Key insight: Something that is (computationally) EASY TO CHECK

Other aspects:

- Interactivity: Proof static, written object? Or dialogue between prover & verifier?
- Correctness: Always 100% correct? Or OK with high probability?

Different combinations possible — yield different results

Today: INTERACTIVE PROOFS

Example: F unsatisfiable CNF formula
- Static (NP-style) proof? Seems hard
- Interactive proof? YES, can be done very efficiently

Interactive proofs important for:
- cryptography (see a little bit next lecture)
- hardness of approximation of NP-complete problems
- indications of non-NP-completeness (hope to mention, but only briefly)

Set-up

Prover — all-powerful, but not trustworthy

Verifier — polynomial-time bounded but can flip coins (privately and hide from prover)

Send messages back and forth

Verifier computes message function f of
- input x,
- randomness r,
- previous messages

Prover computes message function g of
- input x
- previous messages

A *k-ROUND INTERACTION* is a sequence of message strings

$$
\begin{aligned}
a_1 &= f(x, r) \\
a_2 &= g(x, a_1) \\
a_3 &= f(x, r, a_1, a_2) \\
a_4 &= g(x, a_1, a_2, a_3) \\
&\vdots
\end{aligned}
$$

After k rounds verifier announces output $\quad out_f \langle f, g \rangle (x) =$

$$= f(x, r, a_1, a_2, \ldots, a_k) \in \{0, 1\}$$

$$\text{Output} = \begin{cases} 1 & \text{verifier believes } x \in L \\ 0 & \text{verifier does } \underline{\text{not}} \text{ believe } x \in L \end{cases}$$

TRANSCRIPT = sequence of messages sent
random variable over r

DEF 1 (IP)
For $k = k(n)$, (n = input length/size), L is in $\boxed{IP[k]}$ if

$\exists$ PTM V poly-time (in input size $|x|$)
that can have k-round interaction with
prover $P: \{0,1\}^* \to \{0,1\}^*$ such that

COMPLETENESS
$$x \in L \Rightarrow \exists P \quad Pr[out_V \langle V, P \rangle (x) = 1] \geq 2/3$$

SOUNDNESS
$$x \notin L \Rightarrow \forall P' \quad Pr[out_V \langle V, P' \rangle (x) = 1] \leq 1/3$$

where all probabilities are over choice of
V's random string r

$$\boxed{IP} = \bigcup_{c \in \mathbb{N}^+} IP[n^c]$$

Constants are again arbitrary

LEMMA 2 Can set completeness parameter
to $1 - 2^{-n^s}$ and soundness error parameter
to 2^{-n^s} for any constant $s > 0$ without
changing IP.

Proof Boosting as in our error reduction for BPP.

PROP 3 The class of languages in IP decided by deterministic verifier is exactly NP.

Proof. Arora–Barak.

~~~

- Allowing prover to use randomness doesn't help. Can think of probabilistic prover as random choice between deterministic provers. If high success probability on average, then ∃ some deterministic prover with high success probability

- Prover is in principle unbounded – can even solve undecidable problems. But can be shown that given verifier V it is possible to compute optimal prover for x using poly(|x|) space
  Decide probability of acceptance ⇒ shows whether $x \in L$ or not.
  So $IP \subseteq PSPACE$
  [More careful argument needed for a formal proof, of course]

- Possible to replace completeness 2/3 with exactly 1 (perfect completeness)
  [Nontrivial fact]
~~~

- By contrast, reducing soundness error to exactly 0 makes verifier deterministic $\Rightarrow$ collapse to NP
- Randomness hidden from prover – called *private coin* interactive proofs. There are also *public coin* interactive proofs (will see soon)
- In Lem 2, boost success probability by *sequential repetition*. Can also do *parallel repetition* by asking questions in parallel and run protocols in parallel. But then prover can correlate answers to improve chances. However, Prover's probability of winning still decreases quickly
  [Nontrivial – we will skip this]

~~~

Example 4 Interactive proof for GRAPH NON ISOMORPHISM

Two graphs G_1 & G_2 ISOMORPHIC if $\exists\ \pi : V(G_1) \to V(G_2)$ one-to-one and onto s.t. $(u,v) \in E(G_1) \iff (\pi(u), \pi(v)) \in E(G_2)$

Notation $G_1 \cong G_2$

GRAPH ISOMORPHISM $= \{ \langle G_1, G_2 \rangle \mid G_1 \cong G_2 \}$
~~~

GRAPH ISOMORPHISM $\in$ NP
(certificate : π)

Can be solved in ~~quasi~~ polynomial time
$\exp(\log^{O(1)} n)$ [Babai '16]

Even before that was not believed to be NP-complete (unless PH collapses)

Important component in proof of this:

Interactive proof for GRAPH NON ISOMORPHISM (GNI for short)

Protocol for GNI

$V(G) = [n]$

V: Pick $i \in_R \{1,2\}$
Pick random permutation $\pi^* : [n] \to [n]$
Apply π^* to G_i to get graph H
with $V(H) = [n]$
$E(H) = \{(\pi^*(u), \pi^*(v)) \mid (u,v) \in E(G_i)\}$
Send H to P

P: Respond with $j \in \{1,2\}$

V: Accept if $i = j$; reject otherwise

Analysis

$G_1 \not\cong G_2$: Then all-powerful prover can find j for which $H \cong G_j$ and respond with correct answer

$G_1 \cong G_2$ In this case Prover just sees a random permutation of the same graph. Information-theoretically impossible to tell the difference.

For every graph H Prover has fixed answer
But H equally likely for $i=1$ and $i=2$
So Prover correct with probability = 1/2

Repeat protocol twice to get $1/4 < 1/3$

Interesting aspect: What does verifier learn?

Learns whether $G_1 \cong G_2$ or not
Learns nothing beyond this fact

ZERO-KNOWLEDGE PROOFS (kunskapslösa bevis)

What does it mean to "learn nothing"?!

Show that verifier can produce same kind of transcript (with correct probability distribution) in polynomial time without prover

So if this conversation was helpful for something else, then verifier could have talked to him-/herself ...

Useful in cryptography:

Ex Prove you know your password without revealing it.
More about this in a lecture or two...

Public coins

What if prover sees coin flips?
Then verifier need only send coin flips
— prover can compute questions him-/herself

DEF 5 Arthur-Merlin protocols

AM[k] = IP[k] except
- verifier only sends random bits
- can't use any other randomness

Arthur = king
Merlin = court magician

Merlin-Arthur protocols: Merlin goes first

Sometimes notation
AM, AMA, AMAM depending on # rounds

M: This is just NP
A: This is just BPP
AM = BP·NP [requires some thinking]
MA interesting, important class

Standard notation AM = AM[2]

As before, verifier = Arthur announces decision in the end

AM[2k] = deterministic decision based on transcript
AM[2k+1] = randomized computation to reach decision

Some facts (that we won't prove)

LEMMA 6 $\forall k$, k constant, $AM[k] = AM[2]$

THEOREM 7 [Goldwasser, Sipser '87]

For every $k = k(n)$ computable in time $poly(n)$ it holds that

$$IP[k] \subseteq AM[k+2]$$

So having private coins not really that important!

Can show $GNI \in AM[2]$, which in turns indicates (already before [Babai '16]) that GRAPH ISOMORPHISM unlikely to be NP-complete

[Read Sec 8.2 in Arora-Barak if interested — we are not able to cover this in class]

We've argued $NP \overset{(1)}{\subseteq} IP \overset{(2)}{\subseteq} PSPACE$

(1) Probably strict — what about (2)?

What does interaction buy you? Nothing; IP with deterministic verifier $= NP$

What does randomness buy you? Probably nothing; believe $P = BPP$

So what does interaction + randomness buy you? Not too much, right? WRONG!

THEOREM 8 [Lund, Fortnow, Karloff, Nisan '90] [Shamir '90]

$IP = PSPACE$

$IP \subseteq PSPACE$ not hard, as handwaved before

For $PSPACE \subseteq IP$, consider PSPACE-complete language TQBF and show $TQBF \in IP$.

We'll prove something weaker than this, but conveys the main ideas

THEOREM 9 $coNP \subseteq IP$

Want protocol for

$$\overline{3\text{-SAT}} = \{\varphi \mid \varphi \text{ unsatisfiable 3-CNF}\}$$

Design more general protocol for proving number of satisfying assignments

$$\#SAT_D = \{\langle \varphi, K \rangle \mid \varphi \text{ 3-CNF with exactly } K \text{ satisfying assignments}\}$$

Note that $K = 0$ gives $\overline{3\text{-SAT}}$ as special case

Key trick: ARITHMETIZATION of CNF formula

Fix some suitable finite field $\mathbb{F}$ (e.g., computing modulo some prime p)

1 = truth = multiplicative identity of $\mathbb{F}$

0 = falsity = additive identity

$\neg x_i = \overline{x_i}$ true iff $1 - x_i = 1$

$x_i \wedge x_j$ true iff $x_i \cdot x_j = 1$

$x_i \vee x_j$ true iff $1 - (1-x_i)(1-x_j) = 1$

Rewrite every 3-clause

$$C_j = x_i \vee \overline{x_j} \vee x_k$$

as degree-3 polynomial

$$P_j(x_1, \ldots, x_n) = 1 - (1-x_i)\, x_j\, (1-x_k)$$

Evaluates to 1 over 0/1-assignments iff clause true.

Represent $\varphi = \bigwedge_{j=1}^{m} C_j$ as

$$P_\varphi = \prod_{j=1}^{m} P_j(x_1, \ldots, x_n) \qquad (*)$$

Degree $\leq 3m$

P_φ has efficient representation in size $O(m)$

α satisfying assignment to $\varphi \Rightarrow P_\varphi(\alpha) = 1$

α falsifying $-\,''\,-$ $\Rightarrow P_\varphi(\alpha) = 0$

Hence, # satisfying assignments is

$$K = \sum_{\sigma_1 \in \{0,1\}} \cdots \sum_{\sigma_n \in \{0,1\}} P_\varphi(\sigma_1, \ldots, \sigma_n) \qquad (**)$$

Do calculations in $(**)$ modulo some prime $p > 2^n \geq K$ so that answer mod p correct

High-level idea

Input: $\langle \varphi, K \rangle$

Prover sends prime $p \in (2^n, 2^{2n}]$

Verifier checks that p indeed prime
(can be done in poly time)

Then run protocol to check

$$K = \sum_{b_1 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} g(b_1, \ldots, b_n) \qquad (\dagger)$$

for polynomial g with efficient representation so that it can be evaluated in poly time (holds for $g = P_\varphi$)

Observation If we plug in $x_i = b_i$ for $i = 2, \ldots, n$, then $g(x_1, b_2, \ldots, b_n)$ is univariate poly, and so is

$$h(x_1) = \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} g(x_1, b_2, \ldots, b_n) \qquad (\ddagger)$$

The equality $(\dagger)$ holds iff $h(0) + h(1) = K$

So verifier can
(i) ask prover for $h(x_1)$
(ii) check $h(0) + h(1) = K$
(iii) check that prover didn't cheat with h.

Design protocol SumCheck(g, K, n) to check that n-variate polynomial g satisfies $(\dagger)$

SUMCHECK (g, K, n)

V: If $n = 1$ accept if $g(0) + g(1) = K$, otherwise reject.

If $n \geq 2$, ask prover to send (efficient representation of) $h(x_1)$ in $(\star)$

↖ easy; h univariate polynomial

P: Reply with $s(x_1)$

V: Check if $s(0) + s(1) = K$ and reject o/w

Pick random $a \in [0, p-1]$

Let $K' = s(a)$

Let $g' = g(a, x_2, \ldots, x_n)$

Run SUMCHECK $(g', K', n-1)$

Recursive call to protocol checks that prover wasn't cheating but answered $s(x_1) = h(x_1)$ by verifying

$$s(a) = \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} g(a, b_1, \ldots, b_n)$$

But note that a is not $\in \{0,1\}$ but an arbitrary number in $[0, p-1]$!!

LEMMA 10

The protocol SUMCHECK (g, K, n) when run on a degree-d polynomial and computing modulo prime p has
- completeness 1
- soundness error $\leq dn/p$

~~~

In our case
φ has m clauses $\Rightarrow$ P_φ degree $\leq 3m$

m clauses over n variables $\Rightarrow$

$$m \leq \binom{n}{3} 3^3 \leq 27n^3 \quad [\text{why?}]$$

Pick $p > 2^n$

Get soundness error $\leq$

$$\frac{dn}{p} < \frac{81n^4}{2^n} \to 0 \text{ as } n \to \infty$$

So we just need to prove Lemma 10.

This will have to wait till next lecture ...
~~~